BAB II

LANDASAN TEORI

2.1 Penjadwalan Produksi

Menurut Schroeder (1994) penjadwalan adalah keputusan pengalokasian sumber daya yang tersedia dari pengambilan keputusan fasilitas dan perencanaan agregat. Dengan aturan penjadwalan adalah menjadwalkan terlebih dahulu jenis produk yang mempunyai waktu penyelesaian yang terendah. Ini akan membantu untuk memastikan bahwa produk dengan waktu penyelesaian terendah ditempatkan pertama dalam jadwal. Menurut Yamit (2003) penjadwalan adalah gambaran waktu yang diperlukan untuk melaksanakan tugas dengan memperhatikan faktor-faktor sebagai berikut: syarat-syarat tugas, perkiraan permintaan, dan kapasitas yang tersedia. Dari beberapa definisi yang disebutkan dapat disimpulkan bahwa "penjadwalan adalah suatu kegiatan pengalokasian sumber daya, mesin-mesin maupun tenaga kerja untuk melaksanakan tugas dengan mempertimbangkan kapasitas yang tersedia.

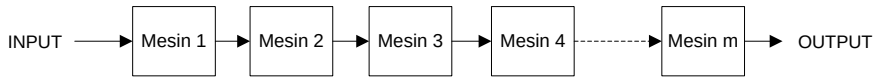
2.1.1 Klasifikasi Penjadwalan

Model penjadwalan yang ada dalam lantai produksi dapat dibagi menjadi beberapa kriteria yaitu sebagai berikut:

1. Berdasarkan mesin yang digunakan dalam proses, penjadwalan produksi dapat diklasifikasikan menjadi:
    1) Penjadwalan pada mesin tunggal
    2) Penjadwalan pada mesin jamak
2. Berdasarkan pola aliran proses, penjadwalan produksi dapat diklasifikasikan sebagai berikut:
    1) Penjadwalan *flowshop*, dimana dijumpai pola aliran dari suatu mesin ke mesin yang lain. Walaupun dalam *flowshop* semua tugas akan mengalir pada jalur produksi yang sama. Macam-macam penjadwalan produksi *flowshop* sebagai berikut:

a. *Pure Flowshop*

Pada penjadwalan *pure flowshop*, semua *job* memiliki jalur produksi yang sama serta tidak ada variasi. Pola semua *job*, setiap operasi dikerjakan pada satu buah mesin dan tidak ada proses ataupun mesin yang dilewati pengerjaan produk.

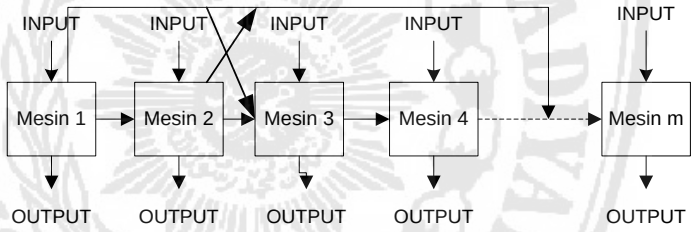


(Sumber: Baker, Kenneth (1974))

Gambar 2.1 Proses *Pure Flowshop*

b. *General Flowshop*

General flowshop merupakan *flowshop* yang memiliki pola aliran proses yang berbeda. Ini disebabkan adanya variasi dalam pengerjaan tugas, sehingga tugas yang datang tidak dikerjakan pada semua mesin.



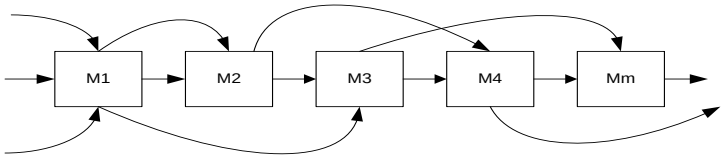
(Sumber: Baker, Kenneth (1974))

Gambar 2.2 Proses *General Flowshop*

Dalam *general flowshop* dibagi menjadi 3 kategori yaitu:

a) *Skip flowshop*

Aliran *job* pada jenis aliran proses ini cenderung melalui urutan proes yang sama, tetapi ada beberapa *job* yang tidak diproses pada mesin-mesin tertentu.

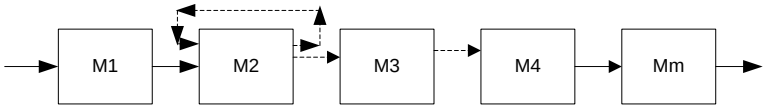


(Sumber: Baker, Kenneth (1974))

Gambar 2.3 Proses *Skip Flowshop*

b) *Reetrant flowshop*

Yaitu proses dimana terdapat penggunaan satu atau beberapa mesin lebih dari sekali dalam membuat produk dimaksud.

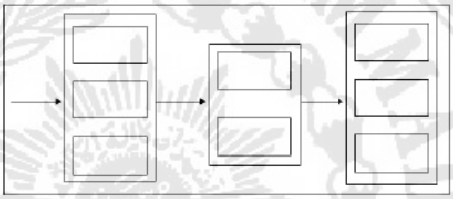


(Sumber: Baker, Kenneth (1974))

Gambar 2.4 Proses *Reetrant flowshop*

c) *Compound flowshop*

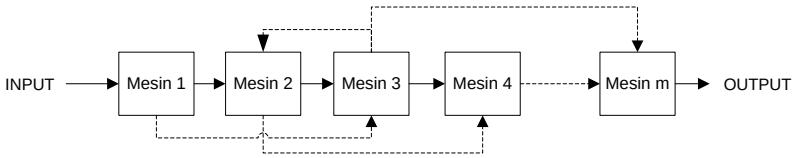
yaitu aliran proses yang membuat kelompok jenis mesin pada setiap tahap prosesnya. Kelompok mesinbiasanya berupa mesin-mesin paralel.



(Sumber: Baker, Kenneth (1974))

Gambar 2.5 Proses Compound Flwoshop

2) Penjadwalan *job shop*, merupakan penjadwalan dimana setiap pekerjaan mempunyai aliran atau rute proses pada tiap mesin yang spesifik, dan mungkin berbeda unutk tiap *job*. Akibat aliran proses yang tidak searah ini maka setiap *job* yang akan diproses, dan yang akan keluar dari sebuah mesin dapat merupakan *job* jadi atau *job* dalam proses.

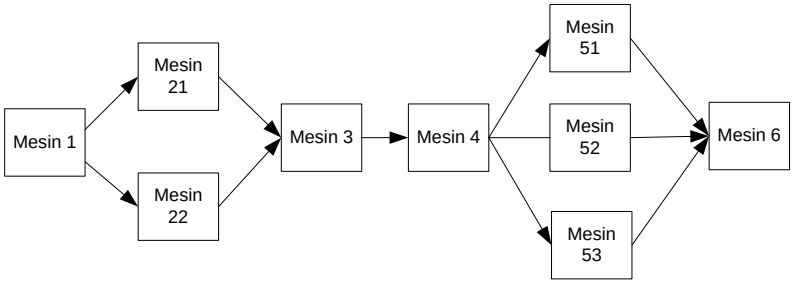


(Sumber: Baker, Kenneth (1974))

Gambar 2.6 Proses Job shop

3) *Flexible flowshop* merupakan rute yang diambil setiap pekerjaan sama seperti yang dilewati aliran proses *flowshop*, akan tetapi dalam prosesnya akan terdapat jumlah mesin yang lebih dari satu tipe dan hanya digunakan

dalam satu pekerjaan saja. Keuntungannya adalah pekerjaan dapat diselesaikan dengan singkat, karena menggunakan mesin lebih dari satu.



(Sumber: Baker, Kenneth (1974))

Gambar 2.7 Proses *Flexible Flowshop*

4) Berdasarkan pola kedatangan *job* , penjadwalan produksi dapat diklasifikasikan menjadi:
   a. Penjadwalan statis, dimana *job* yang datang bersamaan dan siap dikerjakan pada mesin yang menggangur.
   b. Penjadwalan dinamis, dimana kedatangan *job* tidak menentu.

2.1.2 Metode Penjadwalan

Dalam pembuatan jadwal secara keseluruhan sangat sulit dilakukan karena menyangkut aktifitas perusahaan secara keseluruhan. Salah satu kunci keberhasilan untuk meningkatkan efisiensi dalam operasi adalah mampu untuk menyusun jadwal secara efektif. Adapun beberapa metode yang dapat digunakan untuk menyusun jadwal dapat diuraikan sebagai berikut beserta kelebihan dan kekurangannya (Baker, 1974):

1. Metode Heuristik

   Metode heuristik merupakan suatu teknik untuk penyelesaian permasalahan yang tidak menekankan pada pembuktian apakah solusi yang didapatkan adalah benar (pembuktian apakah suatu solusi adalah benar merupakan fokus dari metode penyelesaian analitik), tetapi lebih menekankan pada performa komputasi dan kesederhanaan. Metode heuristik merupakan suatu metode penyelesaian yang menggunakan konsep pendekatan. Metode heuristik yang dapat digunakan salah satunya adalah *branch and bound*.

2. *Integer programming*

   Adalah suatu teknik pemograman untuk menyelesaikan permasalahan penjadwalan dengan menggunakan persamaan-persamaan model matematis beserta kendala-kendala yang dapat mempengaruhi kegiatan penjadwalan. Salah satu metode yang dapat digunakan adalah *mixed integer programming*. Dikatakan sebagai program *mixed integer* karena ada beberapa tetapi tidak semua variabel dibatasi menjadi bilangan bulat.

3. Metaheuristik

   Menurut Talbi (2009) metaheuristik dapat didefinisikan sebagai metode lanjut (*advanced*) berbasis heuristic untuk menyelesaikan persoalan optimisasi secara efficient. Metaheuristik merupakan sebagai metode optimisasi yang dilakukan dengan memperbaiki kandidat penyelesaian secara iteratif sesuai dengan fungsi objektifnya. Metode ini mampu menghasilkan penyelesaian yang baik dalam waktu yang cepat (*acceptable*), tetapi tidak menjamin bahwa penyelesaian yang dihasilkan merupakan penyelesaian terbaik (*optimal*). Metode metaheuristik banyak dipakai dalam optimisasi stokastik (optimisasi stokastim merupakan optimisasi yang memiliki derajat ketidakpastian atau random).

Ada beberapa tujuan yang ingin dicapai dengan dilaksanakannya penjadwalan adalah sebagai berikut (Baker, 1974):

1. Meningkatkan produktivitas mesin, yaitu dengan mengurangi waktu mesin menganggur.
2. Mengurangi persediaan barang setengah jadi dengan mengurangi jumlah rata-rata pekerjaan yang menunggu natrian suatu mesin arena mesin tersebut sibuk.
3. Mengurangi keterlambatan karena telah melampaui batas waktu dengan cara,
   a. Mengurangi maksimum keterlambatan,
   b. Mengurangi jumlah pekerjaan yang terlambat.
4. Meminimasi ongkos produksi.
5. Pemenuhan batas waktu yang telah ditetapkan (*due date*), karena dalam kenyataan apabila terjadi keterlambatan pemenuhan *due date* dapat dikenakan suatu denda (*penalty*).

2.1.3 Ukuran Performansi Penjadwalan

Sebelum membahas kriteria maupun teknik yang digunakan dalam penjadwalan maka harus mengetahui beberapa istilah yang digunakan dalam penjadwalan adalah sebagai berikut (Baker, 1974):

1. Waktu Siap (*ready time*), r_i

   Menunjukkan saat pekerjaan ke-*i* dapat dikerjakan (siap dijadwalkan) *ready time* dapat juga dianggap sebagai waktu kedatangan produk (bahan baku) atau dengan kata lain adalah ketika pekerjaan *j* sampai diperalatan proses atau mesin.

2. Waktu Menunggu (*waiting time*), W_i

   Adalah waktu tunggu pekerjaan *i* dari saat pekerjaan siap dikerjakan sampai saat operasi pendahulu selesai.

3. *Set up Time*

   Adalah waktu yang dibutuhkan untuk kegiatan persiapan sebelum pemrosesan *job* dilaksanakan.

4. *Arrival Time* (a_i)

   Adalah saat *job* mulai berada di *shop floor*.

5. *Delivery Date*

   Adalah saat pengiruman *job* dari *shop floor* proses berikutnya atau ke konsumen.

6. *Processing Time* (t_i)

   Adalah waktu yang dibutuhkan untu mengerjakan suatu pekerjaan. Waktu proses ini, sudah termasuk waktu yang dibutuhkan untuk persiapan dan pengaturan (*setup*) selama proses berlangsung atau merupakan waktu yang diperlukan unutk menyelesaikan suatu operasi, atau proses ke-*i* dari *job* ke-*j*. Waktu proses ini telah mencakup waktu untuk persiapan dan pengaturan proses.

7. *Due Date* (d_i)

   Adalah batas waktu operasi terakhir suatu pekerjaan harus selesai, atau batas waktu penyelesaiaan yang ditentukan unutk *job* ke-*i*.

8. *Slack Time* (SL_i)

   Adalah waktu tersisa yang muncul akibat dari waktu prosesnya lebih kecil dari *due date*-nya.

$$SL_i = d_i - t_i \quad (1)$$

9. *Flow Time* (F_i)

   Adalah waktu yang dibutuhkan oleh suatu pekerjaan dari saat pekerjaan tersebut masuk ke dalam suatu tahap proses sampai pekerjaan yang bersangkutan selesai dikerjakan. Dengan kata lain., *flow time* adalah waktu proses ditambah dengan waktu menunggu sebelum diproses, atau waktu antara *job* ke-*i* siap dikerjakan sampai *job* tersebut diselesaikan,

$$F_i = C_i - r_i \quad (2)$$

10. *Lateness* (L_i)

    Adalah selisih antara *completion time* (C_i) dengan *due date*-nya (d_i). Sutau pekerjaan memiliki *lateness* yang bernilai positif apabila pekerjaa tersebut diselesaikan setelah *due date*-nya, pekerjaan tersebut akan memiliki keterlambatan yang negatif. Sebaliknya jika pekerjaan diselesaikan setelah batas waktunya, pekerjaan tersebut memiliki keterlambatan yang positif, atau besarnya simpangan waktu penyelesaian *job* ke-*i* terhadap *due date* yang telah ditentukan untuk *job* tersebut,

$L_i = C_i - d_i < 0$, saat penyelesaian *job* sebelum batas akhir. (3)

$L_i = C_i - d_i = 0$, saat penyelesaian *job* tepat sesuai batas akhir. (4)

$L_i = C_i - d_i > 0$, saat penyelesaian *job* setelah batas akhir. (5)

11. *Completion time* (C_i)

    Adalah waktu yang dibituhkan untuk menyelesaikan pekerjaan mulai dari saat tersedianya pekerjaan ($t = 0$) sampai pada pekerjaan tersebut selesai dikerjakan, atau menunjukkan rentang waktu sejak pekerjaan pertama mulai dikerjakan sampai proses tersebut selesai,

$$C_i = F_i + r_i \quad (6)$$

12. *Tardiness* (T_i)

    adalah ukuran waktu terlambat yang bernilai positif jika suatu pekerjaan dapat diselesaikan lebih cepat dari *due date*-nya, pekerjaan tersebut akan memiliki

keterlambatan yang negatif. Sebaliknya jika pekerjaan diselesaikan setelah batas waktunya, pekerjaan tersebut memiliki keterlambatan yang positif. Atau besarnya keterlambatan dari *job* j. Terdiness adalah lateness yang berharga positif,

$T_j \geq 0$ jika $L_j \geq 0$

$T_j = 0$ jika $L_j < 0$

$T_i = \max(0, L_i)$

13. *Earliness* (e_j)

    Adalah keterlambatan yang bernilai negatif.

    $e_j \geq 0$ jika $L_j < 0$

    $e_j = 0$ jika $L_j \geq 0$

14. *Makespan* (*M*)

    Adalah total waktu penyelesaian pekerjaan-pekerjaan mulai dari urutan pertama yang dikerjakan pada mesin atau *work center* pertama sampai kepada urutan pekerjaan terakhir pada mesin atau *work center* terakhir.

Ukuran performansi merupakan tujuan dari penjadwalan akan hasil yang diinginkan, (c). Kriteria ukuran performansi yang digunakan untuk mengevaluasi penjadwalan mesin dapat diklarifikasikan menjadi dua yaitu:

1. Kriteria berdasarkan atribut tugas.
    a. *Completion time*, yaitu saat selesai pengerjaan *job* pada suatu stasiun kerja dimana:

    $$C_{maks} = \text{maks}\{c\} \quad (7)$$

    b. *Mean flow time*, merupakan waktu rata-rata dihabiskan pekerjaan *j* dilantai pabrik. *Flow time* adalah selisih *Completion Time* dengan *Ready Time*.

    $$\bar{F} = \frac{1}{n}\sum_{j=1}^{n} F_j \quad (8)$$

    c. *Mean weight flowtime*, memiliki arti yang hampir sam dengan *flowtime*, hanya saja mempertimbangkan prioritas pengerjaan setiap *job* dimana dalam perhitungan sebagai berikut:

$$Fw = \frac{\sum_{j=1}^{n} W_j F_j}{\sum_{j=1}^{n} W_j} \quad (9)$$

d. Minimasi *mean lateness*, yaitu besarnya simpangan maksimum atau selisih waktu penyelesaian seluruh *job* yang dijadwalkan terhadap batas waktu penyelesaian *job* tersebut (*due date*). Lateness bernilai negatif jika waktu penyelesaian *job* lebih awal dari *due date*, dan bernilai positif jika *job* diselesaikan setelah *due date* yang ditentukan untuk *job* tersebut. kriteria ini dipakai jika performansi yang diinginkan adalah pekerjaan ingin diselesaikan lebih awal sebelum *due date*,

$$\bar{L} = \frac{1}{n}\sum_{j=1}^{n} L_j \quad (10)$$

e. Minimasi *maximum lateness*, yaitu besarnya simpangan maksimum, atau selisih waktu penyelesaian seluruh job yang dijadwalkan terhadap batas waktu penyelesaian *job-job* tersebut (*due date*). *Lateness* bernilai negatif jika waktu penyelesaian *job* lebih awal dari *due date*, dan bernilai positif jika *job* diselesaikan detelah *due date* yang ditentukan untuk *job* tersebut. Criteria ini dipakai jika performansi yang diinginkan adalah pekerjaan ingin diselesaikan lebih lambat setelah *due date*,

$$L_{maks} = \max \{L_j\} \quad (11)$$

f. Minimasi maksimasi *tardiness*

$$Tmaks = 1 \leq j^{maks} \leq n^{(Tj)} \quad (12)$$

g. *Mean tardiness*, merupakan rata-rata keterlambatan seluruh *job* yang dijadwalkan yang dapat dihitung dengan rumus sebagai berikut:

$$T = \frac{1}{n}\sum_{j=1}^{n} T_j \text{ dimana } T_j = maks\ (0, L_j) \quad (13)$$

h. *Mean wieght tardiness*, yaitu rata-rata keterlambatan seluruh *job* yang akan dijadwalkan dengan memasukan faktor prioritas oengerjaan masing-masing *job* kedalam perhitungan fungsi tujuannya,

$$Tw = \frac{\sum_{j=1}^{n} W_j T_j}{\sum_{j=1}^{n} W_j} \quad (14)$$

i. Minimasi jumlah job yang terlambat (*number of tardy job*)

Menunjukkan kuantitas *job* yang mengalami keterlambatan,

$$N_t = \sum_{j=1}^{n} N_j \quad (15)$$

Dimana

$N_t = 1$ jika $C_j \geq d_j$

$N_t = 0$ jika $C_j \leq d_j$

2. Kriteria berdasarkan atribut pabrik.

a. Maksimasi utilitas rata-rata mesin, merupakan rasio dari sejumlah wkatu proses yang dibebankan pada mesin dengan rentang waktu untuk menyelesaikan seluruh tugas pada semua mesin,

$$Um = \frac{\sum_{i=n}^{n} t_j}{C_{maks}} \quad (16)$$

b. Minimasi *makespan*, yaitu jangka waktu penyelesaian seluruh *job* yang akan dijadwalkan yang merupakan jumlah dari seluruh proses,

$$Ms = \sum_{k=1}^{n} \sum_{k=1}^{n} tik \quad (17)$$

c. Pemenuhan *due date*, yaitu merupakan penyelesaian pekerjaan sesuai dengan batas waktu yang ditentukan oleh pelanggan dimana harus selalu dilakukan produsen untuk mempertahanan pelanggannya.

2.1.4 Aturan Prioritas Penjadwalan

Aturan prioritas penjadwalan (*priority rute*) yang umumnya digunakan adalah:

1. FCFS (*First Come First Served*)

   Memilih *job* berdasarkab yang datang dahulu, itu yang diproses. Dengan menggunakan aturan ini, proses dikerjakan sesuai dengan urutan ketika *job* tersebut tiba pada fasilitas yang bersangkutan.

2. EDD (*Earliness Due Date*) aturan ini mengurutkan proses dari yang mempunyai *due date* terkecil hingga yang terbesar ($d_1 \leq d_2 \leq ... \leq d_n$). Proses dengan *due date* paling dekat akan diproses terlebih dahulu. Setelah itu penjadwalan dilakukan berdasarkan urutan tersebut.

3. SPT (*Shortest Processing Time*)

   Dengan menggunakan aturan ini, proses dengan waktu operasi yang paling pendek akan dijadwalkan terlebih dahulu. aturan ini sangat sederhana yaitu dengan cara mengurutkan *job* dari yang mempunyai waktu proses terkecil hingga yang terbesar ($t_1 \leq t_2 \leq ... \leq t_n$). Setelah itu penjadwalan dilakukan berdasarkan urutan tersebut.

4. LPT (*Longest Processing Time*)

   Dengan menggunakan aturan ini, proses dengan waktu operasi yang paling panjang akan dijadwalkan terlebih dahulu. aturan ini jiga sangat sederhana yaitu dengan cara mengurutkan *job* dari yang mempunyai waktu proses terbesar hingga yang terkecil ($t_1 \geq t_2 \geq ... \geq t_n$). Setelah itu penjadwalan dilakukan berdasarkan urutan tersebut.

5. LS (*Least Slack*)

   Aturan ini menerapkan *job* dengan *slack* yang lebih kecil terlebih dahulu. slack adalah sisa waktu yang ada antara due date dengan waktu ketika *job* itu sesuai. Atutan ini mengurutkan *job* dari yang mempunyai waktu slack terkecil hingga yang tebesar ($SL_1 \leq SL_2 \leq ... SL_n$). Setelah itu penjadwalan dilakukan berdasarkan urutan tersebut.

6. CR (*Critical Ratio*)

   Urutan *job* berdasarkan CR terkecil (mengurangi *lateness*). Ratio kritis merupakan suatu nomor indeks yang dihitung dengan membagi waktu yang tersisa sampai tanggal jatuh tempo dengan sisa waktu kerja. selayaknya berlawanan dengan algoritma prioritas, ratio kritis adalah dinamis dan sangat mudah untuk diperbaharui. Ratio kritis cenderung lebih bagus dari FCFS, SPT, EDD dan LPT pada kriteria rata-rata keterlambatan kerja.

$$CR = \frac{due\ date - now}{reamaining\ lead\ time} \quad (18)$$

2.2 *Integer Programming*

Menurut Tarliah & Dimyati (1994) program bilangan bulat atau *integer programming* (IP) adalah bentuk lain dari programa linier (LP) dimana asumsi divisibilitasnya melemah atau hilang sama sekali. Bentuk ini muncul karena dalam kenyataannya tidak semua variabel keputusan dapat berupa bilangan pecahan. Asumsi divisibilitas melemah, artinya sebagian dari nilai variabel keputusan harus berupa bilangan bulat (*integer*) dan sebagian bisa berupa bilangan pecahan. Persoalan IP dimana hanya sebagian dari variabel keputusannya yang harus *integer* disebut sebagai persoalan IP campuran (*Mixed Integer Programming*).

Sebagai contoh perhatikan formulasi berikut:

Maksimumkan: $z = 3x_1 + 2x_2$

berdasarkan:

$x_1 + x_2 \leq 6$

$x_1, \quad x_2 \geq 0$; x_1 integer (x_2 tidak harus integer)

Apabila seluruh variabel keputusan dari suatu persoalan programa linier (LP) harus berharga integer, maka persoalan tersebut disebut sebagai persoalan programa bilangan bulat (IP) murni. Dalam hal ini, asumsi divisibitas dari LP hilang sama sekali.

Sebagai contoh, perhatikan formulasi berikut ini:

Maksimumkan: $z = 3x_1 + 2x_2$

Berdasarkan:

$x_1 + x_2 \leq 6$

$x_1, x_2 \quad \geq 0$; x_1, x_2 integer

secara umum, model persoalan IP dapat diformulasikan sebagai berikut:

maks/Min: $z = \sum_{j=1}^{n} c_j x_j$

berdasarkan:

$$\sum_{j=1}^{n} a_{ij} \quad x_j \begin{matrix} \leq \\ = \\ \geq \end{matrix} b_i,\ i = 1, \ldots, m$$

$x_j \geq 0, j = 1, \ldots, n$

x_j integer untuk j = 1, 2, ..., p (p ≤ n)

Ada beberapa bentuk umum model *integer programming*, yaitu (Yamit, 2003):

1. *Integer Linear Programming* atau *Integer Programming* (IP) adalah sebagian atau semua variabel keputusan berbentuk integer atau bilangan bulat.

   Optimumkan (maksimum atau minimum)

   $Z = f(x_1, x_2, ..., x_n)$

   Dengan kendala

   $(x_1, x_2, ..., x_n) \leq$ atau $\geq b_i$

   Dan $x_j \geq 0$, untuk i = 1, 2, ..., m

   x_j integer untuk j = 1, 2, ..., n

2. *All Integer programming* atau *Pure Integer programming* adalah jika semua variabel keputusan berbentuk integer.

   Optimumkan (maksimum atau minimum)

   $Z = 3x_1 + 2x_2$

   Dengan kendala

   $x_1 \leq 2$

   $x_2 \leq 2$

   $x_1 + x_2 \leq 3{,}5$

   $x_1, x_2 \geq 0$ dan integer

3. *Mixed Integer Programming* (MIP) adalah jika beberapa variabel keputusan berbentuk integer.

   Optimumkan (maksimum atau minimum)

   $Z = 4x_1 + 3x_2$

   Dengan kendala

   $2x_j + 2x_2 \leq 2$

   $x_2 \leq 1$

   $x_1 + x_2 \leq 1$

   $x_1, x_2 \geq 0$ dan x_2 integer

4. *Binary Variabel* adalah jika variabel keputusan tertentu berbentuk integer tetapi hanya memiliki sepasang nilai 0 atau 1.

   Optimumkan (maksimum atau minimum)

$Z = 40x_1 + 30x_2$

Dengan kendala

$2x_1 + 4x_2 \leq 60$

$5x_2 \leq 50$

$3x_1 + 2x_2 \leq 30$

$x_1 \geq 0$

$x_2 = 0$ atau 1

5. *Binary Integer Programming* atau 0-1 *Integer Programming* adalah jika semua variabel keputusan berbentuk integer dan memiliki sepasang nilai 0 atau 1.

   Maksimum $Z = 100x_1 + 75x_2$

   Dengan kendala

   $4x_1 + 2x_2 \leq 100$

   $2x_1 + x_2 \leq 50$

   $x_1, \quad x_2 = 0$ atau 1

2.2.1 Penyelesaian MIP Dengan Teknik *Branch and Bound*

Menurut Hiller (2012) mendeskripsikan suatu dasar algoritma *branch and bound* untuk memecahkan masalah MIP yang dengan berbagai penyempurnaan sehingga memberikan pendekatan masalah MIP. Langkah-langkah penyelesaian MIP dengan teknik *branch and bound* :

1. Percabangan: diantara sub masalah yang tersisa, pilih salah satu sub masalah yang baru saja dibuat. Putuskan ikatan yang memiliki batas yang lebih besar. Di antara variabel terbatas bilangan bulat yang memiliki nilai non bilangan bulat dalam solusi optimal untuk relaksasi LP dari sub masalah, pilih yang pertama dalam pengurutan alami variabel menjadi percabangan. Biarkan x_j menjadi variabel ini dan x_j^* nilainya dalam solusi ini. Cabang dari node untuk sub masalah untuk membuat dua sub masalah baru dengan menambahkan batasan masing-masing $x_j \leq [x_j^*]$ dan $x_j \geq [x_j^*]+1$.

2. *Bounding*: untuk setiap sub maslaha baru, dapatkan ikatannya dengan menerapkan metode simpleks (untuk metode simpleks ganda saat reptimisasi) ke relaksasi LP dan menggunakan nilai *Z* untuk solusi optimal yang dihasilkan.

3. Fathoming: untuk setiap sub masalah baru, terapkan tiga tes fathoming yang diberikan dibawah ini, dan buang sub masalah yang dipahami oleh salah satu tes.

Tes 1: yang terikat $\leq Z^*$, dimana Z^* adalah nilai Z untuk pemegang jabatab saat ini.

Tes 2: relaksasi LP tidak memiliki solusi yang layak.

Tes 3: solusi optimal untuk relaksasi LP memiliki nilai integer untuk variabel terbatas bilangan bulat. Jika solusi ini lebih baik daripada *incumbent*, itu menjadi *incumbent* baru dan tes 1 diterapkan kembali ke semua sub masalah yang tidak disamakan dengan Z^*.

Uji optimalisasi: berhenti ketika tidak ada sub masalah yang tersisa; pemegang *incumbent* optimal, jika tidak; melakukan iterasi lain.

Contoh MIP: kita sekarang akan mengilustrasikan algoritma ini dengan menerapkannya ke masalah MIP berikut.

Minimasi $z = 4x_1 - 2x_2 + 7x_3 - x_4$

Kendala:

$$x_1 + 5x_3 \leq 10$$
$$x_1 + x_2 - x_3 \leq 1$$
$$6x_1 - 5x_2 \leq 0$$
$$-x_1 + 2x_3 - 2x_4 \leq 3$$

Dan

$x_j \geq 0$, untuk $j = 1, 2, 3, 4$.

x_j adalah integer, untuk $j = 1, 2, 3$.

Note: bahwa jumlah variabel terbatas bilangan bulat adalah $I = 3$, jadi x_4 adalah satu-satunya variabel kontinyu.

Inisialisasi: setelah pengaturan $Z^* = -\sim$, kami membentuk LP relaksasi masalah ini dengan menghapus himpunan kendala yang x_j adalah bilangan bulat untuk $j = 1, 2, 3$. Menerapkan metode simpleks untuk relaksasi LP ini menghasilakn solusi optimal dibawah ini:

LP relaksasi seluruh masalah: $(x_1, x_2, x_3, x_4) = \left(\frac{5}{4}, \frac{3}{2}, \frac{7}{4}, 0\right)$, dengan $Z = 14\frac{1}{4}$.

Karena memilki solusi yang layak dan solusi optimal ini memiliki bilangan non integer untuk variabel terbatas bilangan bulat, seluruh masalah tidak dipahami, sehingga algoritma berlanjut dengan iterasi penuh pertama dibawah ini:

Iterasi 1: dalam solusi optimal relaksasi LP, variabel bilangan bulat terbatas pertama yang memiliki nilai non bilangan bulat adalah $x_1 = \frac{5}{4}$, jadi x_1 menjadi variabel percabangan. Bercabang dari semua node (semua solusi yang layak) dengan variabel percabangan ini kemudian membuat dua sub masalah berikut:

Sub masalah 1: masalah asli ditambahkan kendala tambahan

$x_1 \leq 1$

sub masalah 2: masalah asli ditambah kendala tambahan

$x_1 \geq 2$

Menghapus set kendala integer lagi dan memecahkan relaksasi LP yang dihasilkan dari kedua sub masalah ini menghasilkan hasil sebagai berikut:

LP relaksasi sub masalah 1: $(x_1, x_2, x_3, x_4) = \left(1, \frac{6}{5}, \frac{9}{5}, 0\right)$, dengan $Z = 14\frac{1}{5}$.

Terikat untuk sub masalah 1: $Z \leq 14\frac{1}{5}$. LP relaksasi sub masalah 2P tidak ada solusi yang layak.

Hasil ini untuk sub masalah 2 berarti bahwa itu diukur dengan tes 2. Namun, seperti untuk seluruh masalah, sub masalah 1 gagal semua tes yang diukur. Hasil ini dirangkum dalam pohon solusi.

Iterasi 2: dengan hanya satu sub masalah yang tersisa, sesuai dengan $x_1 \leq 1$, percabangan berikut adalah dari node ini. memeriksa LP relaksasi solusi optimal yang diberikan dibawah ini, node ini mengungkapkan bahwa variabel percabangan adalah x_2, karena $x_2 = \frac{6}{5}$ adalah variabel pembatas integer pertama yang memiliki nilai no integer. Menambahkan alah satu batasan $x_2 \leq 1$ atau $x_2 \geq 2$ kemudian membuat dua sub masalah baru sebagai berikut:

Sub masalah 3: masalah asli ditambah kendala tambahan $x_1 \leq 1$, $x_2 \leq 1$.

Sub maslaah 4: masalah asli ditambah kendala tambahan $x_1 \leq 1$, $x_2 \geq 2$.

Menyelesaikan relaksasi LP merak memberikan hasil berikut:

LP relaksasi sub maslah 3 : (x_1, x_2, x_3, x_4) = $\left(\frac{5}{6}, 1, \frac{11}{6}, 0\right)$, dengan $Z = 14\frac{1}{6}$,

Terikat untuk sub masalah 3: $Z \leq 14\frac{1}{6}$

LP relaksasi sub maslah 4 : (x_1, x_2, x_3, x_4) = $\left(\frac{5}{6}, 2, \frac{11}{6}, 0\right)$, dengan $Z = 12\frac{1}{6}$,

Terikat untuk sub maslaah 4: $Z \leq 12\frac{1}{6}$.

Karena kedua solusi ada (solusi yang layak) dan memiliki nilai non integer untuk variabel terbatas integer, tidak ada sub masalah yang dipahami, (tes 1 masih tidak operasional, karena $Z^* = 1 \sim$ sampai *incumbent* pertama ditemukan).

Iterasi 3: dengan dua sub masalah tersisa (3 dan 4) yang dibuat secara bersamaan, yang dengan batas besar (sub masalah 3, dengan $14\frac{1}{6} > 12\frac{1}{6}$) dipilih untuk percabangan berikut. Karena $x_1 = \frac{5}{6}$ memiliki nilai non integer dalam solusi optimal untu relaksasi LP sub masalah ini, x_1 menjadi variabel percabangan. (perhatikan bahwa x_1 sekarang adalah variabel percabangan berulang, karena itu juga dipilih di iterasi 1). Ini mengarah ke sub masalah baru berikut:

Sub masalah 5: masalah asli ditambah kendala tambahan $x_1 \leq 1$, $x_2 \leq 1$, $x_1 = 0$ (jadi $x_1 = 0$)

Sub masalah 6: masalah asli ditambah kendala tambahan $x_1 \leq 1$, $x_2 \leq 1$, $x_1 = 1$ (jadi $x_1 = 1$)

Hasil dari penyelesaian relaksasi LP mereka ditambahkan dibawah ini:

LP relaksasi sub masalah 5: (x_1, x_2, x_3, x_4) = $\left(0,0,2,\frac{1}{2}\right)$, dengan $Z = 13\frac{1}{2}$,

Terikat untuk sub masalah 5: $Z \leq 13\frac{1}{2}$,

LP relaksasi sub masalah 6: tidak ada solusi yang layak.

Sub masalah 6 segera diukur dengan tes 2. Namun, perhatikan bahwa sub masalah 5 juga dapat dipahami. Tes 3 berlalu karena solusi optimal untuk relaksasi LP memiliki nilai integer ($x_1 = 0$, $x_2 = 0$, $x_3 = 0$) untuk ketiga variabel integer terbatas. (tidak masalah bahwa x_3 $\frac{1}{2}$, karena x_4 tidak dibatasi secara integer). Solusi layak untuk masalah asli ini menajadi *incumbent* pertama kami:

Incumbent = $\left(0,0,2,\frac{1}{2}\right)$, dengan *Z* *= $13\frac{1}{2}$,

Menggunakan *Z* ini * untuk menerapkan kembali tes 1 ke satu-satunya sub masalah (sub masalah 4) yang berhasil, karena terikat pada $12\frac{1}{6}, \leq$ z *. iterasi ini telah berhasil dalam memahami sub masalah, sehingga petahanan saat ini optimal. solusi optimal = (0,0,2, $\frac{1}{2}$) dengan z = $13\frac{1}{2}$.

2.3 Penelitian Terdahulu

Ronconi & Birgin (2010) mengembangkan model *Mixed Integer programming* pada penjadwalan *flowshop* untuk menyelesaikan problem minimasi *total earliness* dan *tardiness* dengan mengkombinasi model formulasi dari Wilson's (1989).

Fungsi tujuan

$$\text{Minimize } \sum_{j=1}^{n} E_j + T_j \tag{19}$$

Kendala-kendala:

$$T_j \geq C_{jm} - \sum_{i=1}^{n} x_{ij} d_i \qquad j = 1, 2, ..., n \tag{20}$$

$$E_j \geq \sum_{i=1}^{n} x_{ij} d_i - C_{jm} \qquad j = 1, 2, ..., n \tag{21}$$

$$C_{jm} = S_{jm} + \sum_{i=1}^{n} x_{ij} p_{im} \qquad j = 1, 2, ..., n \tag{22}$$

$$S_{j+1,k} \geq S_{jk} + \sum_{i=1}^{n} x_{ij} p_{ik} \qquad j = 1, 2, ..., n\text{-}1,\ k = 1, 2, ..., m \tag{23}$$

$$S_{j,k+1} \geq S_{jk} + \sum_{i=1}^{n} x_{ij} p_{ik} \qquad j = 1, 2, ..., n,\ k = 1, 2, ..., m\text{-}1 \tag{24}$$

$$S_{11} \geq 0 \tag{25}$$

$$\sum_{i=1}^{n} x_{ij} = 1 \qquad j = 1, 2, ..., n \tag{26}$$

$$\sum_{j=1}^{n} x_{ij} = 1 \qquad j = 1, 2, ..., n \tag{27}$$

Parameter:

j = posisi proses penjadwalan, *j* = 1, 2, ..., *n* ; posisi yang memungkinkan;

i = nomor *job*, *i* = 1, 2, ..., *n* ; *n* adalah jumlah *job*;

k = nomor mesin *k* = 1, 2, ..., *m* ; *m* adalah jumlah mesin dalam proses;

P_{im} = waktu proses *job i* pada mesin *k* ;

d_i = due date *job i* ;

variabel:

E_j = *earliness* pada *job j* ;

T_j = keterlambatan pada *job j* ;

S_{jk} = waktu mulai dari *job j* pada mesin *k* ;

P_{ik} = waktu proses *job i* pada mesin *k* ;

x_{ij} = bernilai 1 apabila *job i* berada pada *job j* ;

C_{jm} = waktu penyelesaian *job j* pada mesin *m* ;

S_{jm} = waktu mulai *job j* pada mesin *m* ;

Fungsi tujuan (19) minimasi *total earliness* dan *tardiness*. Kendala (20) bernilai non-negatif $T_j \geq 0$ sesuai dengan rumus tardiness $T_j = \max\{C_{jm} - \sum_{i=1}^{n} x_{ij} d_i, 0\}$ dan memberikan nilai keterlambatan setiap *job*. Kendala (21) bernilai non-negatif $E_j \geq 0$ sesuai dengan rumus earliness $E_j = max\{\sum_{i=1}^{n} x_{ij} d_i - C_{jm}, 0\}$. Kendala (22) menunjukkan waktu penyelesaian *job* dimesin terakhir bukan variabel namun nilai tengah yang digunakan untuk menyederhanakan pada kendala (20) dan (21). Kendala ini memberikan waktu penyelesaian setiap *job* pada mesin terakhir sebagai fungsi waktu mulai dan waktu prosesnya. Kendala (23) menyatakan bahwa antara awal kali pekerjaan berturut-turut pada mesin, harus ada cukup waktu untuk yang pertama (dari dua pekerjaan) yang diproses. Kendala (24) menunjukkan bahwa antara waktu mulai pekerjaan pada dua berturut-turut mesin, harus ada cukup waktu agar pekerjaan dapat diproses pada mesin pertama. Kendala (25) waktu mulai suatu *job* pertama dimesin pertama harus bernilai non-negatif $S_{11} \geq 0$. Kendala (26) dan (27) suatu *job* hanya akan dialokasikan ke posisi urutan dan setiap posisi urutan hanya dilakukan untuk satu *job* saja.